Un algoritmo evolutivo híbrido para la planificación minera por extracción subterránea

Carlos Grandón, Teddy Alfaro y Michael Moossen
Laboratorio de Modelos y Métodos Cuantitativos
Departamento de Informática
Universidad Técnica Federico Santa María
Avda. España 1680, Vaparaíso, Chile
{cobra, teddy, mmoossen}@labmc.inf.utfsm.cl

Resumen

El problema de planificación minera constituye un campo importante de investigación en el sector minero nacional, debido a sus características de "intratabilidad" al intentar ser resuelto por métodos tradicionales. Pertenece al conjunto de problemas de optimización combinatoria, con crecimiento exponencial de su espacio de búsqueda. Debido a esto, es que se han propuesto métodos alternativos para encontrar una solución aproximada al problema. Este documento plantea la aplicación de un algoritmo evolutivo híbrido, combinado con heurística GRASP para la planificación minera.

Palabras Clave: Planificación minera, Algoritmo evolutivo híbrido, GRASP, reparación, operadores especializados.

1. Introducción

El problema de secuenciamiento en la extracción de mineral constituye un campo importante de investigación en el sector minero nacional. Esto se debe a los grandes montos de inversión que involucra y plazos de realización de proyectos que fluctúan entre los 20 y 25 años. Actualmente, los estudios más especializados en este tema se aplican a minas a tajo abierto existentes en el país. Estas minas poseen un procedimiento de extracción desde las capas superiores hacia las inferiores, aumentando progresivamente la zona de explotación.

A diferencia de este tipo de minas, las de extracción subterránea poseen un procedimiento de extracción que parte en el corazón de la mina, eliminando capas desde los niveles inferiores hacia los superiores, provocando un conjunto de restricciones adicionales al problema. Estas restricciones le otorgan al problema sus características de "intratabilidad" al intentar ser resuelto por métodos tradicionales.

En la búsqueda de nuevas estrategias, se han propuesto modelos específicos como el mostrado en [4], que intentan rescatar las características relevantes del problema. Por otro lado, en su resolución, se han planteado algoritmos basados en heurística GRASP [1] con buenos resultados.

El presente documento plantea la aplicación de un algoritmo evolutivo híbrido para la planificación minera. Además, se utiliza la metaheurística GRASP en la creación de poblaciones iniciales, y operadores especializados para la evolución de la población.

En la siguiente sección, se explican las características principales del problema, sus restricciones y objetivo. En la sección 3 se presenta un modelo del problema, orientado a la aplicación de heurísticas. La sección 4 presenta el algoritmo propuesto, los parámetros involucrados y el tipo de heurística aplicada. La sección 5 presenta los casos de prueba considerados en el estudio del algoritmo, mientras que la sección 6 presenta los resultados obtenidos, incluyendo el resultado específico de la mina real. Por último, la sección 7 presenta las conclusiones y trabajos futuros, seguidas por la bibliografía.

2. Descripción del Problema

La planificación minera se define como el conjunto de actividades que realiza una empresa con el objetivo de determinar los recursos económicos que posee en sus yacimientos y establecer una estrategia de extracción y de proceso del material, la cual genere un valor económico lo más alto posible [2].
El problema de determinar la planificación óptima, es un problema de alta complejidad. Esto, debido al horizonte de tiempo involucrado (generalmente 20 años) y a la gran cantidad de alternativas posibles, tanto tecnológicas como operativas.
El proceso minero, en su totalidad, comprende cuatro grandes etapas:

Labores de Geología: En esta etapa se estudia la distribución de mineral en los yacimientos, utilizando distintas técnicas. Entre ellas, las geoestadísticas son las más usadas.

Labores de Extracción: Es el proceso de extracción del material de la mina, como también de construcción de túneles de acceso a ella.

Tareas de Concentración: Es el proceso posterior a la extracción. Una vez que se tiene el material en bruto, es necesario tratarlo para extraer su riqueza, para lo cual se hace un proceso previo antes de mandar el material a la fundición. Esto se hace mediante una serie de procesos, realizados tanto dentro como fuera de la mina.

Proceso de Fundición: Corresponde a la última etapa, en la cual se desea obtener el mineral en altos niveles de leyes.

Este trabajo se centra en las labores de extracción del mineral, basados en la distribución de mineral obtenida en la etapa previa, considerando la información geológica como válida y determinística. Dada la información geológica, se define la riqueza de una zona, de acuerdo a la cantidad de mineral que esta posea. Esta riqueza se determina a partir de los costos totales de explotación, la ley de material y el precio del mineral.
El objetivo de este trabajo es determinar la secuencia de extracción que signifique un mayor beneficio económico. Dado que este beneficio está relacionado con los precios de venta de los minerales, estos precios se consideran determinísticos. La forma de determinar el beneficio económico es el *Valor Actual Neto* (VAN), cuya definición exacta se presenta en la sección 3.3. Esto incluye la variable tiempo en la decisión de extracción.
Además, la tarea de extracción de material, está sujeta a algunas restricciones, tanto lógicas como tecnológicas. Estas restricciones se mencionan a continuación:

Extracción hacia arriba: Dado que el método de explotación consiste en extraer el material que se encuentra arriba de un túnel construido, lo que implica un costo de construcción y habilitación del tunel, la extracción es de forma ascendente.

Capacidad: Existe una capacidad máxima de extracción anual debido a consideraciones tecnológicas.

Subsidencia: El lugar donde se está extrayendo material, define un cono superior a este. Así, el material incluído en este cono no puede ser extraído posteriormente. Esto, debido al peligro de desplome de material.

Dadas las consideraciones anteriores, se puede definir el problema como la determinación del plan de extracción que maximice los beneficios económicos, utilizando como medida de este, el VAN.

A continuación una ilustración esquemática de una mina:

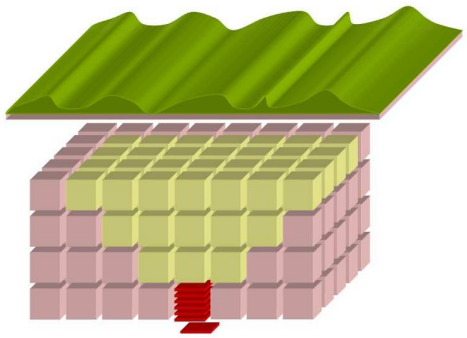

Figura 1: Esquema de un sector de una Mina

3. Modelo Matemático

El primer paso que se realiza para definir el problema, es discretizar el yacimiento en sectores. Para cada uno de estos sectores, se tiene información obtenida por los estudios geológicos, tal como ley de minerales, densidad y características geológicas. A partir de esta información, se puede determinar la riqueza que tiene cada uno de estos sectores y su ubicación dentro del yacimiento.
Cada sector contiene cierta cantidad de bloques, y cada bloque puede ser extraído en su totalidad o parcialmente en porcentajes. Para esto, se introduce el concepto de capas dentro de los bloques, representando cada capa el 10 % de un bloque. Así, para cada sector, el objetivo es determinar la secuencia de extracción óptima de capas de bloques del sector, considerando que las capas de un mismo bloque deben ser extraídas en orden, es decir, no se puede extraer el material al medio del bloque, sin haber extraído el material anterior. Además, se tiene que tomar en cuenta un costo de habilitación de cada bloque para poder extraer mineral de él.

3.1. Parámetros

p	Horizonte de tiempo en años.
e	Cantidad de capas máxima que se pueden extraer por año.
C	Costo de habilitación de un bloque.
n	Cantidad de bloques del sector.
r	Tasa de descuento anual.
U_i	Utilidad del bloque i, $i \in \{1 \dots n\}$.

3.2. Variables

$x_{i,t}$ Porcentaje de extracción del bloque i en el tiempo t, en múltiplos de 10 %.
$i \in \{1 \dots n\}$, $t \in \{1 \dots e * p\}$

$H_{i,t}$ El bloque i es habilitado en el tiempo t.

$$H_{i,t} = \begin{cases} 1 & \text{El bloque } i \text{ es habilitado en el tiempo } t \\ 0 & \text{e.t.o.c.} \end{cases}$$

$i \in \{1 \dots n\}$, $t \in \{1 \dots e * p\}$

3.3. Función Objetivo

$$\text{Maximizar } VAN = \sum_{i,t} \frac{1}{(1+r)^t} \left(U_i x_{i,t} - C H_{i,t}\right) \tag{1}$$
$$i \in \{1 \dots n\},\ t \in \{1 \dots e * p\}$$

3.4. Restricciones

No se puede extraer más del 100 % de un bloque

$$\forall i \in \{1 \dots n\} \quad \sum_{t=1}^{e*p} x_{i,t} \leq 1 \tag{2}$$

Cada bloque es habilitado una sola vez

$$\forall i \in \{1 \dots n\} \quad \sum_{t=1}^{e*p} H_{i,t} \leq 1 \tag{3}$$

Sólo bloques habilitados pueden ser explotados

$$\forall i \in \{1 \dots n\} \quad \forall t \in \{1 \dots e * p\} \quad x_{i,t} \leq \sum_{1<s<t} H_{i,s} \tag{4}$$

Restricciones de Subsidencia

$$\forall i \in \{1 \dots n\} \quad \forall s \geq t \quad \forall t \in \{1 \dots e * p\} \quad H_{i,t} + H_{i,s} \leq 1 \tag{5}$$
$$\forall j \in S(i) \quad H_{i,t} + x_{j,s} \leq 1$$

Donde, $S(i)$ corresponde al conjunto de bloques que pertenecen al cono de subsidencia del bloque i.

4. Algoritmo Evolutivo Propuesto

La estructura utilizada en el algoritmo evolutivo se presenta a continuación:

1. **Inicialización:** Creación de la población inicial. Los individuos son obtenidos por medio de un procedimiento semi-aleatorio o mediante un algoritmo GRASP.
2. **Evaluación:** Resultado de la función de evaluación de cada individuo.
3. **Elitismo:** Seleccionar para la nueva población el mejor individuo de la población actual.
4. **Selección:** Es utilizado el sistema "roulette whell" [5] para seleccionar los individuos que completan la población. Para esto, se aplica la transformación "Truncación Sigma" citado en [5], a la función de evaluación definida en la sección 4.2.
5. **Mutación:** Mutar al azar una proporción m de la población a base de un operador exploratorio.
6. **Cruzamiento:** Cruzar al azar una proporción c de la población, utilizando un operador especializado cruzamiento.
7. **Condición de Término:** Volver hasta el punto 2, hasta completar i iteraciones:

4.1. Representación

La representación utilizada es una lista de genes (x, y, z), es decir, una lista de bloques (con x, y y z coordenadas espaciales del bloque), de los cuales se extrae una capa a la vez. En caso de extraer más de una capa del mismo bloque, se repite el gen, la cantidad de veces necesarias.

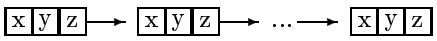

Figura 2: Representación de un Individuo

4.2. Función de Evaluación

La función de evaluación utilizada para evaluar a cada individuo, corresponde al cálculo del *Valor Actual Neto*, tal como se definió en la sección 3.3

4.3. Generación de Población Inicial Semi-Aleatoria

Corresponde a un algoritmo que se basa en las características físicas del problema de la extración de minas. Una restricción muy fuerte es el cono de subsidencia. Para abstraerse de tal problema, este generador de población inicial, realiza extracciones promedio por cada nivel Z, ya que si dos bloques pertenecen al mismo nivel Z, nunca se bloquearán. A continuación se presenta el algoritmo base de este operador.

4.3.1. Pseudo-Código

Paso 0: Inicialización
Calcular la cantidad de capas a extraer por nivel

Paso 1: Ciclo de Construcción por nivel Z
Para cada nivel en el eje Z, en orden descendente, hacer:
Mientras no se extraigan las capas necesarias para este nivel, hacer:
Seleccionar aleatoriamente coordenadas x e y
Seleccionar aleatoriamente cantidad de capas c a extraer
Insertar extracción x, y, z como un nuevo gen, c veces en el individuo
Actualizar capas libres del bloque
Fin Mientras
fin Para

En primer lugar, se obtiene el número de capas que se necesita extraer por nivel. Luego, recorriendo los niveles Z en orden descendentes, se extraen las capas necesarias por cada nivel.

Como desventaja se puede decir que los individuos son creados siguiendo un mismo patrón, por lo que carecen de variedad. Por otro lado, como ventaja se tiene que es sumamente rápido, y siempre se obtienen soluciones factibles.

4.4. GRASP para Soluciones Iniciales

A continuación, se describe el algoritmo GRASP utilizado en la generación de la población inicial. Este algoritmo no respeta explícitamente la restricción del cono de subsidencia, por lo cual los individuos generados deben ser luego reparados. Se basa en el cálculo del *máximo promedio ponderado por capas(MPPC)* de cada bloque, como se explica a continuación:
Sean

$p \in N$:	Número de períodos
$e \in N$:	Extracciones por Período
$dx \in N$:	Dimensión de la mina en el eje X
$dy \in N$:	Dimensión de la mina en el eje Y
$dz \in N$:	Dimensión de la mina en el eje Z
$dc \in N$:	Cantidad de capas por Bloque
$U(x,y,z,c) \in R$:	La utilidad neta al comenzar el primer período de extraer la Capa c, del Bloque ubicado en las coordenadas (x,y,z)
$ch \in R_+$:	Costo de habilitación de un Bloque
$c(x,y,z,t) \in \{0 \dots dc\}$	La cantidad de capas ya extraídas del bloque en las coordenadas (x,y,z).

Es necesario usar un mecanismo especial para comparar riquezas de bloques habilitados y no habilitados, debido al costo de habilitación. Para seleccionar el *mejor* bloque a extraer posible en un instante dado, en cada iteración, se calcula α para cada bloque que aún tenga capas sin extraer, donde α es el *MPPC* del bloque, en base al cual se comparan las distintas posibilidades. α se define como:

- Si el bloque ha sido previamente habilitado:

$$\alpha(x,y,z,t) = \max_{h \in \{[c(x,y,z,t)+1],\dots,dc\}} \left\{ \sum_{i=c(x,y,z,t)+1}^{h} \frac{U(x,y,z,i)}{h - c(x,y,z,t)} \right\} \tag{6}$$

- e.t.o.c.:

$$\alpha(x,y,z,t) = \max_{h \in \{[c(x,y,z,t)+1],\dots,dc\}} \left\{ \left(\sum_{i=c(x,y,z,t)+1}^{h} \frac{U(x,y,z,i)}{h - c(x,y,z,t)} \right) - \frac{ch}{h - c(x,y,z,t)} \right\} \tag{7}$$

Luego, se elige aleatoriamente un bloque de entre los diez mejores(con este valor se obtuvieron los mejores resultados experimentales) con respecto a α y se extraen la cantidad de capas, que llevó al cálculo de $\alpha(h - c(x,y,z,t))$.

4.4.1. Seudo-Código

Paso 0: Inicialización

por cada bloque:
  calcular α
  insertar α, el bloque y la cantidad de capas en una lista L
fin por cada
ordenar L en forma descendente, según α

Paso 1: Selección

entre los diez primeros(mejores) elementos de L, seleccionar uno al azar
extraer las capas del bloque seleccionado, según como se obtuvo α.
reducir la cantidad de capas disponibles de este bloque.

Paso 2: Actualización

remover el elemento usado de L
si aún quedan capas por extraer en este bloque:
calcular nuevamente α para este bloque.
insertar α junto al bloque y la cantidad de capas en la lista ordenada.
fin si

Paso 3: Iteración

si es necesario extraer más capas para completar el individuo:
volver al paso 1.
fin si

Los individuos así obtenidos, no son necesariamente factibles, ya que en ningún momento se toma explicítamente en consideración la restricción del cono de subsidencia. Por lo que es necesario aplicar un algoritmo de reparación, ya que se desea trabajar con individuos factibles.

4.4.2. Reparador de Subsidencia

El reparador de subsidencia pretende convertir individuos infactibles en factibles. Para ello, se considera la definición del cono de subsidencia: *Todo bloque j perteneciente al cono superior en 45º con respecto a la normal de un bloque i, quedará inhabilitado para extracción toda vez que el bloque i sea habilitado.* Luego para todo bloque i, con coordenadas (x_i, y_i, z_i), el cono de subsidencia queda determinado por todo bloque j que cumple con:

$$\mid x_j - x_i \mid \leq z_j - z_i \qquad \wedge \qquad \mid y_j - y_i \mid \leq z_j - z_i \tag{8}$$

Dada esto, se define una relación de orden como: *Un bloque j es mayor que un bloque i si y sólo si j pertenece al cono de subsidencia de i.* Es decir:

$$j > i \iff [\ |x_j - x_i| \leq z_j - z_i\ ] \quad \wedge \quad [\ |y_j - y_i| \leq z_j - z_i\ ] \quad \forall \quad j \neq i \tag{9}$$

Este operador posee la propiedad de transición dado que:

$$si \quad b_i > b_j \quad \wedge \quad b_j > b_k \quad \Longrightarrow \quad b_i > b_k \tag{10}$$

De esta forma basta realizar un ordenamiento ascendente de cualquier individuo para obtener soluciones que respeten las restricciones de subsidencia.

4.4.3. Seudo-Código

```
por cada gen:
    por cada gen posterior, en la secuencia de extracción:
        si el primer bloque bloquea al segundo:
            intercambiar los bloques.
        fin si
    fin por cada
fin por cada
```

La idea se presenta en la siguiente figura:

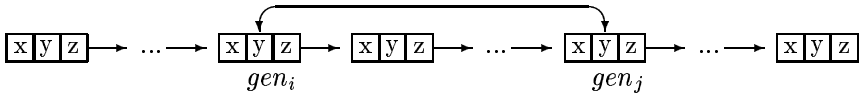

Figura 3: Algoritmo Reparador

En esta figura, el algoritmo está revisando el gen i. Para cada gen j posterior en la secuencia de extracción, se comprueba que el gen i no bloquee al gen j. Si este es el caso, se intercambian los genes y se prosigue con el siguiente.

4.5. Mutación de Exploración Aleatoria

El objetivo de esta mutación es explorar diferentes sectores del espacio de búsqueda. Como se muestra en la figura 4, el algoritmo selecciona un punto aleatorio del individuo original y cambia su información por valores escogidos aleatoriamente. Si las nuevas coordenadas pertenecen a un bloque extraído, el nuevo individuo extraerá una capa adicional de éste. Por el contrario, si las coordenadas no pertenecen a un bloque extraído, el individuo habilitará un nuevo bloque de la mina.

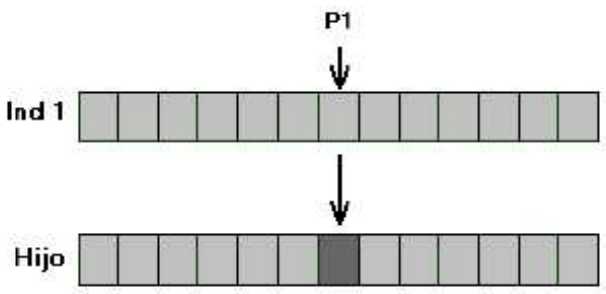

Figura 4: Mutación de exploración aleatoria

Esta mutación permite agregar variabilidad a la población, intentando evitar quedar estancados en óptimos locales.

4.6. Cruzamiento Especializado

Un cruzamiento tradicional resulta inaplicable directamente, ya que requiría de reparaciones y reordenamientos. Debido a ello se diseñó un operador especializado, el cual con la información de dos padres genera un hijo que consta de una etapa de recombinación y otra de optimización. Este cruzamiento fue concebido con la idea de aprovechar de mejor manera la información contenida en los padres. El algoritmo desarrollado para este operador se presenta a continuación.

4.6.1. Seudo-Código

Paso 0: Inicialización

Construir estructura intermedia $p1p2$ de las capas extraídas por los padres $p1$ y $p2$
Calcular los MPPC de cada bloque de $p1p2$

Paso 1: Ciclo de Construcción

Mientras $p1p2$ no sea vacío, hacer:
  Establecer como gen_n la mejor extracción promedio de $p1p2$
  Mientras gen_n no se encuentre en el cono de subsidencia del gen del hijo, hacer:
    Pasar al siguiente gen del hijo
  Fin Mientras
  Borrar gen gen_n de la estructura $p1p2$
  Insertar gen_n en la posición actual
  Actualizar las capas libres del bloque
Fin Mientras

Paso 2: Etapa de Corrección de Sobre-Explotación

Mientras las capas extraídas sea más que las necesarias
  Para cada gen del hijo, hacer:
    Si la utilidad de la extracción es menor que el promedio menos la desviación estándar:
      Eliminar el gen
      Reajustar la consistencia de precedencia en la extracción de capas

Fin Si
Fin Para
fin mientras

En primer lugar, se construye una estructura intermedia que mantiene la información de los bloques y las capas que fueron explotadas por ambos padres. Luego, se calculan las MPPC de cada bloque, se selecciona la mejor extracción promedio, la cual es borrada de la estructura intermedia (ver figura 5), se actualiza la capa libre del bloque (si es que quedan) y son recalculadas las MPPC. Esto corresponde a la etapa de recombinación.

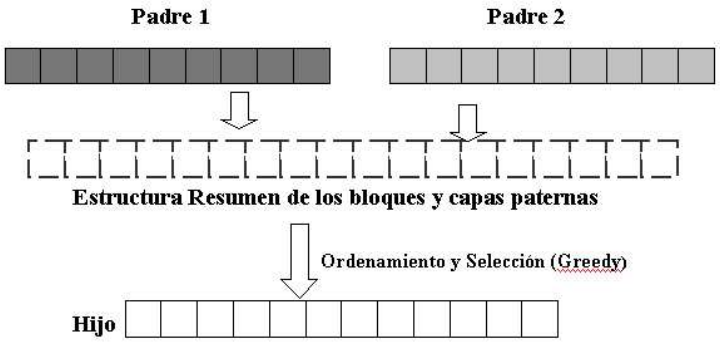

Figura 5: Cruzamiento Recombinatorio-Optimizador

La mejor extracción promedio seleccionada es insertada en el hijo utilizando el siguiente algoritmo que se inicia desde su gen inicial hasta el final:

1. Si el cromosoma hijo está vacío, considerar la extracción como primer gen.
2. Sino, Hacer desde el gen inicial hasta el gen final:
   - Si la extracción no se encuentra en el cono de subsidencia del gen, ir al siguiente gen.
   - Sino, insertar la extracción inmediatamente anterior al gen que lo bloquea.
3. Si no hay más genes, insertar extracción como nuevo gen final.

Y esto corresponde a la etapa de construcción optimizada.
El método de inserción se basa en la propiedad de transitividad de la relación de orden definida en 10.
Recorrer el hijo desde el gen inicial hasta el gen final, asegura una tendencia a que las MPPC sean explotadas lo más tempranamente posible para ser un aporte más significativo a la función de evaluación (en el cálculo del VAN).
En el presente trabajo este operador trabajó sobre individuos factibles, pero debe considerarse que debido a su diseño presenta un potencial para trabajar también sobre individuos infactibles.

4.7. Parámetros del Algoritmo Evolutivo

Para este problema en particular se utilizaron los siguientes parámetros:

- Tam_{Pob}: 20, tamaño de la población. Tamaño propuesto en [5] y además por limitante de capacidad de cómputo.
- P_{cru}: 0.8, probabilidad de Cruzamiento Recombinatorio. La idea es que este operador genético sea altamente utilizado, para así explotar a gran escala la información contenida en los individuos.
- P_{mut}: 0.2, probabilidad de mutación. Se utilizó un valor pequeño de probabilidad, para conceder una cuota considerable de exploración de bloques nuevos.

5. Casos de Prueba

Los casos de prueba analizados utilizan los siguientes parámetros:

- Tasa de interés del 10 % anual
- 10 capas del 10 % cada una por bloque
- Costo de habilitación de un bloque de 8.0
- Capacidad para extraer 40 capas por período y un tiempo de extracción de 20 períodos.

Se ha trabajado con distintos casos de prueba, tanto ficticios como reales. Los diez primeros casos son benchmarks disponibles en `http://www.labmc.inf.utfsm.cl/`, y el último se basa en datos reales de la mina "El Teniente". Para cada caso de prueba, se presenta un resumen de sus características más importantes. En primer lugar, el tipo de problema y su dimensión. Luego, se hace un estudio del rango en que deben estar las soluciones aceptables, para lo cual se definen dos conceptos:

Cota Inferior Aceptable(*CIA*): La idea de esta medición es encontrar un mínimo aceptable para una solución del problema. Para esto se construyó un algoritmo determinista que busca los 80 bloques con más riqueza de la mina. Luego se ordenan por su ubicación en el eje Z, y se extraen de arriba hacia abajo, manteniendo la restricción de subsidencia. Así se obtiene una solución factible, que se usará como cota inferior para el análisis de los resultados obtenidos.

Cota Superior Teórica(*CST*): Para encontrar una cota superior, se extraen los 80 bloques que resultan de agrupar las 800 capas individuales con mayor riqueza en 80 bloques ficticios, que luego son extraídos en orden descendente en riqueza. En general, esto no produce un individuo factible, pero sí se demuestra que ningún individuo (factible o no) puede superar esta cota.

Por último, se indica el resultado máximo obtenido con la heurística evolutiva.

Problema	**Tipo**		**Dimensiones**	
a1	Totalmente Aleatorio		18 x 4 x 10	
Cotas	**Inferior:**	7383.77	**Superior:**	11174.37
Resultado	**V.A.N.:**	8214.79	**Habilitaciones:**	219

Problema	**Tipo**		**Dimensiones**	
a6	Totalmente Aleatorio		9 x 8 x 10	
Cotas	**Inferior:**	7590.87	**Superior:**	11204.29
Resultado	**V.A.N.:**	8367.94	**Habilitaciones:**	208

Problema	**Tipo**		**Dimensiones**	
a11	Totalmente Aleatorio		18 x 8 x 5	
Cotas	**Inferior:**	7570.82	**Superior:**	11219.68
Resultado	**V.A.N.:**	8307.37	**Habilitaciones:**	203

Problema	**Tipo**		**Dimensiones**	
a16	Riquezas al Final		18 x 8 x 5	
Cotas	**Inferior:**	7659.25	**Superior:**	11173.57
Resultado	**V.A.N.:**	8451.14	**Habilitaciones:**	195

Problema	**Tipo**		**Dimensiones**	
a21	Riquezas al Medio		6 x 6 x 20	
Cotas	**Inferior:**	7522.54	**Superior:**	11555.03
Resultado	**V.A.N.:**	8190.25	**Habilitaciones:**	113

Problema	**Tipo**		**Dimensiones**	
a26	Riquezas al Final		6 x 6 x 20	
Cotas	**Inferior:**	6248.40	**Superior:**	11358.12
Resultado	**V.A.N.:**	10173.02	**Habilitaciones:**	215

Problema	Tipo	Dimensiones
a31	Riquezas en los Bordes	4 x 18 x 10
Cotas	**Inferior:** 11417.06	**Superior:** 11621.93
Resultado	**V.A.N.:** 11417.06	**Habilitaciones:** 80

Problema	Tipo	Dimensiones
a36	Riquezas al Medio	8 x 9 x 10
Cotas	**Inferior:** 10795.53	**Superior:** 11558.76
Resultado	**V.A.N.:** 10812.92	**Habilitaciones:** 97

Problema	Tipo	Dimensiones
a41	Riquezas por Capas	9 x 40 x 2
Cotas	**Inferior:** 11137.48	**Superior:** 11536.77
Resultado	**V.A.N.:** 11158.76	**Habilitaciones:** 84

Problema	Tipo	Dimensiones
a46	Riquezas al Final y al Medio	9 x 8 x 10
Cotas	**Inferior:** 9955.10	**Superior:** 11203.40
Resultado	**V.A.N.:** 10051.12	**Habilitaciones:** 107

Problema	Tipo	Dimensiones
real	Mina "El Teniente"	18 x 20 x 2
Cotas	**Inferior:** 21.9623	**Superior:** 722.50
Resultado	**V.A.N.:** 240.38	**Habilitaciones:** 112

6. Resultados

A continuación, se presenta con mayor detalle los resultados obtenidos para el problema real de la mina "El Teniente".

6.1. Población Inicial

El siguiente gráfico presenta una población inicial generada con el algoritmo GRASP y luego reparada mediante el algoritmo reparador presentado, para el problema real de la mina "El Teniente":

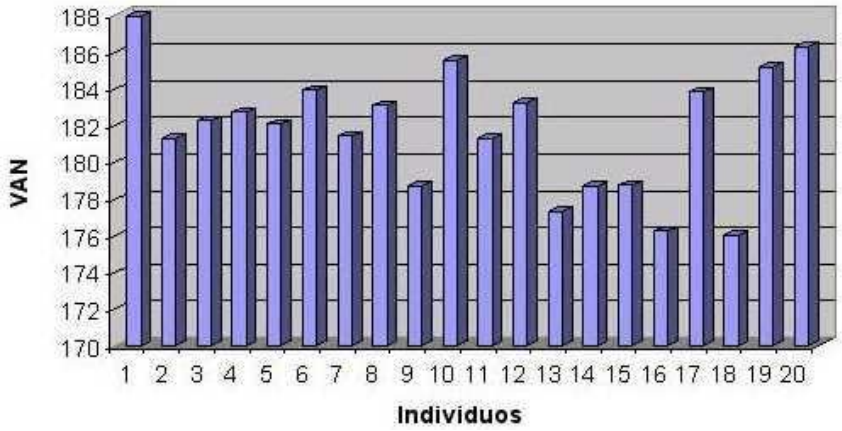

Figura 6: Población inicial generada con GRASP basado en MPPC

Se puede apreciar la buena calidad de estas soluciones iniciales si se comparan con las obtenidas por el algoritmo semi-aleatorio, que se presentan a continuación:

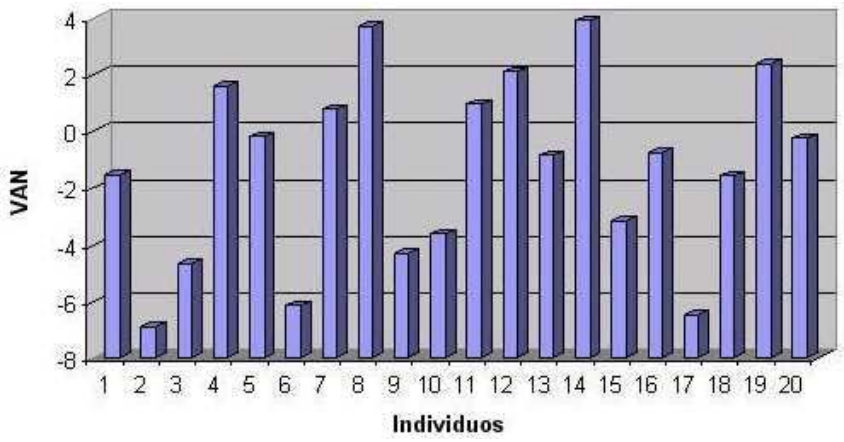

Figura 7: Población inicial generada semialeatoriamente

6.2. Evolución Genética

A continuación se presenta un gráfico que muestra la evolución del mejor individuo durante la ejecución del algoritmo evolutivo, comprobándose el buen comportamiento de la heurística, en el sentido de que el algoritmo parece no estancarse fácilmente en un óptimo local, debido principalmente a la componente exploradora del algoritmo, mejorando iteración tras iteración los mejores individuos, gracias a la componente explotadora del algoritmo:

Figura 8: Evolución del mejor individuo en el tiempo

Estadísticas sobre el tiempo de ejecución por iteración (en un AthlonXP 1.6Ghz con 256Mb RAM):

Mínimo	1.00 seg.
Máximo	3.55 seg.
Promedio	1.88 seg.
Desviación	0.86 seg.
Total (mil iteraciones)	1877.10 seg.

7. Conclusiones

Se planteó un algoritmo evolutivo híbrido para la planificación minera. La combinación de heurísticas expuesta mostró un desempeño superior respecto a los trabajos previos mostrados en [3] y [1]. En todos los casos de prueba analizados, el desempeño del algoritmo, en relación a la cota superior teórica, superó por un margen de 1 % y hasta

10 % los resultados previos. Para el problema real, los resultados obtenidos superan en más del 20 % la cercanía a la cota superior. El mejor valor inicial propuesto (72.95) no alcanza el 30 % del valor final obtenido al aplicar este nuevo algoritmo (240.38).

Por otro lado, es de suma utilidad realizar un análisis previo, tanto del tipo de problema a tratar como de la instancia específica y de este modo, fijar metas de rendimiento acordes. Por ejemplo, en los problemas a31, a36 y a41, la diferencia entre la cota inferior aceptable y la cota superior teórica es de 2 %, 7 % y 3 % respectivamente. Esto sugiere utilizar el algoritmo de cota inferior aceptable como solución del problema, puesto que el costo de mejorarla sobrepasa el beneficio real esperado. Por el contrario, para instancias en donde la diferencia de las cotas es mayor (45 % para **a26** y 97 % para la mina **real**), las ventajas del algoritmo se hacen evidentes con un mejoramiento de hasta 30 % respecto a la cota superior teórica.

Como un aporte adicional, la relación de orden definida en la sección 4.4.2 permite abstraer el problema de subsidencia en la formulación de individuos candidatos a solución, ya que provee una herramienta para transformar individuos infactibles en factibles. Dadas la características de la relación, un operador de reparación, en el peor de los casos, añadirá una complejidad $O(n \log n)$. Con ello, la reparación de individuos no representa un costo elevado para el algoritmo.

Por último, la clave de los buenos resultados obtenidos por el algoritmo se basa en la aplicación de la heurística MPPC planteada en la sección 4.4. Si bien no es posible asegurar la obtención del óptimo, debido a las restricciones de subsidencia y a la influencia del tiempo en la evaluación de las capas, un algoritmo GRASP basado en esta heurística mostró soluciones que son hasta dos órdenes de magnitud mayores que las soluciones generadas en forma semi-aleatoria. Al ser implementada en los operadores (particularmente el operador de cruzamiento) mostró una tendencia hacia buenas soluciones.

8. Trabajo Futuro

Como trabajo futuro se pueden incorporar mecanismos de control de parámetros. Estos trabajos serían una interesante experiencia, ya que pueden significar un gran aporte al algoritmo evolutivo presentado. Además, los buenos resultados obtenidos con el cruzamiento especializado, despierta la inquietud en el estudio de otros operadores especializados que se basen en las características intrínsecas del problema tratado. Por último, no se puede dejar de mencionar un estudio del potencial aún no explorado que presentan los algoritmos: GRASP, reparador basado en una relación de orden, y el cruzamiento especializado.

Referencias

[1] Carlos Grandón. Un algoritmo GRASP aplicado al problema de secuenciamiento de extracción de mineral en la mina "El Teniente". Technical report, Universidad Técnica Federico Santa María, 2002.

[2] Riff M.C. Resumen del problema de planificación minera en mina "El Teniente". Technical report, Universidad Técnica Federico Santa María, 2001.

[3] Moossen Michael. Problema de macrosecuenciamiento de extracción de mineral en la mina "El Teniente". Technical report, Universidad Técnica Federico Santa María, 2002.

[4] Morales Varela Nelson. Revisión del modelo de macrosecuenciamiento. Technical report, Santiago, Octubre, 2001.

[5] Michalewicz Zbigniew. *Genetic Algorithms + Data Estructures = Evolution Programs.* Springer series Artificial Intelligence, 1992.